Group technology in production management : the short horizon planning level

Hervé Garcia, Jean-Marie Proth

INRIA

CENTRE DE ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél. (3) 954 90 20

Rapports de Recherche

N° 376

GROUP TECHNOLOGY IN PRODUCTION MANAGEMENT : THE SHORT HORIZON PLANNING LEVEL

Hervé GARCIA
Jean Marie PROTH

Mars 1985

GROUP TECHNOLOGY IN PRODUCTION MANAGEMENT :

THE SHORT HORIZON PLANNING LEVEL

Software TEC - G1

Hervé GARCIA* and Jean Marie PROTH**

PROJET SAGEP

* Laboratoire d'Automatique de Metz, IUT, Ile de Saulcy.

** INRIA, Rocquencourt, B.P.105, LE CHESNAY Cedex, tél.(3)954 90 20.
INRIA-CIRIL, Château du Montet, Rue du Doyen Roublot, 54500 VANDOEUVRE, tél.(8)355 15 45.

This work is partially supported by the "Agence de l'Informatique".

ABSTRACT

We consider a set of parts divided into subsets called part types, determined in such a way that the parts belonging to the same part type are manufactured using the same sequence of tasks (i.e. the same working process). We are looking for a partition of the set of part types in subsets called part families, and for a partition of the set of tasks in subsets called production subsystems defined as follows : 1) the number of part families and the number of production subsystems are equal 2) one (and only one) production subsystem corresponds to each part family 3) one (and only one) part family corresponds to each production subsystem 4) the previous partitions minimize the number of tasks performed in a production subsystem different from the one which corresponds to the part family containing the part involved. We give a fast algorithm which leads to a good solution depending on the initial set of part families. We also propose an algorithm to find a "good" initial set of part families.

RESUME

Nous considérons un ensemble de pièces divisé en sous-ensembles appelés types de pièces et determinés de manière que les pièces qui appartiennent au même type de pièces soient fabriquées en subissant la même séquence de tâches (i.e. la même gamme de fabrication). Nous cherchons à partitionner l'ensemble des types de pièces en familles de pièces et l'ensemble des tâches en sous-systèmes de production tels que les deux partitions contiennent le même nombre d'éléments, qu'il existe une application biunivoque entre l'ensemble des familles de pièces et l'ensemble des sous-systèmes de production et que ces partitions minimisent le nombre de tâches accomplies dans un sous-système de production différent du sous-système correspondant à la famille de pièces de la pièce concernée. Nous donnons un algorithme rapide qui conduit à une bonne solution qui dépend de l'ensemble initial de familles de pièces. Nous proposons également un algorithme qui conduit à un "bon" ensemble initial de familles de pièces.

I. INTRODUCTION

In this paper, we call working process a sequence of tasks to perform in order to manufacture a part. We say that two parts belong to the same part type if their working processes are the same. Note that we don't take the processing times into account.

We are looking for a partition of the set of part types in subsets called part families, and for a partition of the set of tasks in subsets called production subsystems defined as follows : 1) the number of part families and the number of production subsystems are equal 2) one (and only one) production subsystem corresponds to each part family 3) one (and only one) part family corresponds to each production subsystem 4) these partitions minimize the number of tasks performed in a production subsystem different from the one which corresponds to the part family containing the part involved.

The objective is then to obtain production subsystems which are as independant as possible at the point of view of the production management, according to the set of parts to produce : following the above definition, each production subsystem can be considered approximately as an independant production system dedicated to perform the parts belonging to the corresponding part family. This reduce the size of the scheduling problem. The case of parts which visit production subsystems different from the ones corresponding to their part families is taken into account at the second step of the management process.

This paper is divided into five paragraphs. The problem is set in paragraph 2. In the paragraph 3, we give an algorithm to obtain a good solution of the problem starting from an initial partition of the set of part types. In the paragraph 4, we show how to obtain a good initial partition of the set of part types. Finally, the paragraph 5 gives an example with 60 part types and 12 tasks.

II. SETTING OF THE PROBLEM

We consider a set X of n part types and a set Y of m tasks.
We define the matrix :

$$A = [a_{i,j}],\ i=1,2,\ldots,n \text{ and } j=1,2,\ldots,m$$

as follows :

$$a_{i,j} = \begin{cases} 1 \text{ if the working process of a part belonging to the part type i contains the task j.} \\ 0 \text{ if not.} \end{cases} \quad (1)$$

Let p be the number of part families and of production subsystems to consider. This integer is chosen by the user.

If $\mathscr{X} = \{X_1,\ldots,X_p\}$ and $\mathscr{Y} = \{Y_1,\ldots,Y_p\}$ are respectively a partition of X and a partition of Y, we define the sets $W_{r,s}$, r=1,...,p and s=1,...,p as follows :

$$W_{r,s} = \{(i,j) \in (1,\ldots,n)x(1,\ldots,m)/i \in X_r \text{ and } j \in Y_s\} \quad (2)$$

For a given value of p, $\mathscr{X}^* = \{X_1^*,\ldots,X_p^*\}$ and $\mathscr{Y}^* = \{Y_1^*,\ldots,Y_p^*\}$ are said to be the optimal solution of the problem if they verify :

$$\sum_{r=1}^{p} \sum_{(i,j)\in W^*_{r,r}} u_i a_{i,j} + \sum_{r=1}^{p} \sum_{\substack{s=1 \\ s\neq r}}^{p} \sum_{(i,j)\in W^*_{r,s}} u_i(1-a_{i,j}) =$$

$$= \operatorname{Max}_{\mathscr{W}\in\mathscr{D}} \left[\sum_{r=1}^{p} \sum_{(i,j)\in W_{r,r}} u_i a_{i,j} + \sum_{r=1}^{p} \sum_{\substack{s=1 \\ r\neq s}}^{p} \sum_{(i,j)\in W_{r,s}} u_i(1-a_{i,j}) \right] \quad (3)$$

where :

u_i is the weight of the part type i (for instance, the proportion of parts of type i to produce during a given period)

$W^*_{r,s}$, r=1,2,...,p and s=1,2,...,p are built using (2), starting from $\mathcal{X}^*$ and $\mathcal{Y}^*$.

$\mathcal{W} = \{W_{r,s}\}_{r=1,..,p;s=1,..,p}$

$\mathcal{D}$ is the set of admissible partitions $\mathcal{W}$ (i.e. the partitions of $(1,...,n)x(1,...,m)$ which have p^2 éléments).

Note that the maximal value of this criterion is $m \sum_{i=1}^{n} u_i$.

We also can set the problem as follows :

Starting from matrix A, build a matrix B by permutation of the rows and the columns of A in order to obtain p diagonal blocks $E_1^*,..,E_p^*$ such that :

$$\sum_{(i,j)\in \bigcup_{r=1}^{p} E_r^*} u_i b_{i,j} + \sum_{(i,j)\notin \bigcup_{r=1}^{p} E_r^*} u_i(1-b_{i,j}) \qquad (4)$$

where :

$B = (b_{i,j})$; i=1,2,...,n and j=1,2,...,m.

u_i= is the weight of the part type which corresponds to the row i of the matrix B.

III. A GOOD SOLUTION STARTING FROM A PARTITION OF THE PART TYPES

III.1. A basic result

We first prove the following result.

THEOREM I

Let $\mathcal{X}^k = \{X_1^k, X_2^k, \ldots, X_p^k\}$ and $\mathcal{Y}^k = \{Y_1^k, Y_2^k, \ldots, Y_p^k\}$ be a partition of X and Y respectively.

1. For $j=1,2,\ldots,m$, we consider :

$$\left|\begin{array}{l} a_r^k(j) = \sum_{i\in X_r^k} u_i a_{i,j} + \sum_{\substack{i=1 \\ i\notin X_r^k}}^{n} u_i(1-a_{i,j}) \text{ for } r=1,..,p. \\ \\ r^k(j) \text{ is the integer which verifies :} \\ \\ a^k_{r^k(j)}(j) = \underset{r=1,..,p}{\text{Max}}\ a_r^k(j) \\ \\ \text{We assign the task } j \text{ to the production subsystem } Y^{k+1}_{r^k(j)} \end{array}\right. \qquad (5)$$

Finally, we denote by $\mathcal{Y}^{k+1}$ the partition :

$$\mathcal{Y}^{k+1} = \{Y_1^{k+1}, Y_2^{k+1}, \ldots, Y_p^{k+1}\}$$

(some of the production subsystems Y_r^{k+1} may be empty).

2. For $i=1,2,\ldots,n$, we consider :

$$\left|\begin{array}{l} b_r^k(i) = \sum_{j\in Y_r^{k+1}} a_{i,j} + \sum_{\substack{j=1 \\ j\notin Y_r^{k+1}}}^{m} (1-a_{i,j}) \text{ for } r=1,..,p. \\ \\ r^k(i) \text{ is the integer which verifies :} \\ \\ b^k_{r^k(i)}(i) = \underset{r=1,..,p}{\text{Max}}\ b_r^k(i) \\ \\ \text{We assign the part type } i \text{ to the part family } X^{k+1}_{r^k(i)} \end{array}\right. \qquad (6)$$

Finally, we denote by $\mathcal{X}^{k+1}$ the partition :

$$\mathcal{X}^{k+1} = \{X_1^{k+1}, X_2^{k+1}, \ldots, X_p^{k+1}\}$$

Then, either the solution $(\mathscr{X}^{k+1}, \mathscr{Y}^{k+1})$ is better than the solution $(\mathscr{X}^{k}, \mathscr{Y}^{k})$, or the value of the criterion (see(3)) is the same for both solutions.

Proof :

1. Let us first consider the way to obtain $\mathscr{Y}^{k+1}$ starting from $\mathscr{X}^{k}$(see (5)). We see that the partition $\mathscr{Y}^{k+1}$ is one of the partitions which lead to the greatest value of the criterion (see(3)) knowing $\mathscr{X}^{k}$. Then the value of the criterion for $(\mathscr{Y}^{k+1}, \mathscr{X}^{k})$ is greater than or equal to the value of the criterion for $(\mathscr{Y}^{k}, \mathscr{X}^{k})$.

2. Let us now consider (6). We see that the partition $\mathscr{X}^{k+1}$ is one of the partitions which lead to the greatest value of the criterion knowing $\mathscr{Y}^{k+1}$. Then the value of the criterion for $(\mathscr{Y}^{k+1}, \mathscr{X}^{k+1})$ is greater than or equal to tha value of the criterion for $(\mathscr{Y}^{k+1}, \mathscr{X}^{k})$.

3. Finally, the value of the criterion for $(\mathscr{Y}^{k+1}, \mathscr{X}^{k+1})$ is greater than or equal to the value of the criterion for $(\mathscr{Y}^{k}, \mathscr{X}^{k})$. □

III.2. The algorithm

We suppose that we know a partition of the part types(i.e. a set of part families) denoted by $\mathscr{X}^{0}$. We first compute $\mathscr{Y}^{0}$ using (5) with k=0, then $\mathscr{X}^{1}$ using (6) with k=0, then $\mathscr{Y}^{1}$ using (5) with k=1, then $\mathscr{X}^{2}$ using (6) with k=1 and so on.

We stop the computation when two consecutive solutions lead to the same value for the criterion.

Remark :

This algorithm leads to a solution which depends on the initial set of part families.

For instance, consider the matrix A (fig.1).

	Y_1			Y_2				
	1	1	1	1	0	0	0	0
	1	1	1	1	1	0	0	0
X_1	1	1	1	1	0	0	0	0
	1	0	1	0	0	0	1	1
	1	1	0	0	0	1	1	0
	0	0	0	0	1	1	1	1
X_2	0	0	0	0	1	1	1	1
	0	0	0	0	0	1	1	1
	0	0	0	1	0	1	1	1
			Y_1^*			Y_2^*		

X_1^* : rows 1–3; X_2^* : rows 4–9

Fig.1

Let be :

$\mathcal{Y} = \{Y_1, Y_2\}$, $\mathcal{X} = \{X_1, X_2\}$

$\mathcal{Y}^* = \{Y_1^*, Y_2^*\}$, $\mathcal{X}^* = \{X_1^*, X_2^*\}$

The solution $(\mathcal{X}, \mathcal{Y})$ is stable. It means that applying (5) with $\mathcal{X}$ leads to $\mathcal{Y}$ and applying (6) with $\mathcal{Y}$ leads to $\mathcal{X}$. The value of the criterion is 57 for this solution. The criterion takes the value 60 for the solution $(\mathcal{X}^*, \mathcal{Y}^*)$: therefore $(\mathcal{X}, \mathcal{Y})$ is not optimal.

A consequence of the previous remark is that we have to execute a number of trials to be sure to obtain a solution close to the optimal one.

We now give an algorithm which leads to an initial set of part families which seems to be a good initial set.

IV. THE INITIAL SET OF PART FAMILIES

Each part family is an element of $\mathbb{R}^m$. A distance d is given on $\mathbb{R}^m$. We denote by x_i, i=1,2,...,n, the part families, and u_i the weight of x_i.

The goal consists in partitioning $E = \{x_1, x_2, \ldots, x_n\}$ in p subsets. Two elements of the same subset are close in the sense of the distance d.

We try to minimize the sum of the inertia of the subsets, i.e. :

$$\sum_{k=1}^{p} \sum_{i/x_i \in e^k} u_i d^2(x_i, y^k) \qquad (7)$$

where :

y^k is the inertia center of e^k, $k=1,2,\ldots,p$

and : $\{e^1, e^2, \ldots, e^p\}$ is the partition.

THEOREM II

Let $x_1^0, x_2^0, \ldots, x_p^0$ ($p \leq n$) be some points of $\mathbb{R}^m$, belonging to E or not.

We consider the following algorithm, called A1 :

1. Set k=0.
2. Compute a partition $e_1^k, e_2^k, \ldots, e_p^k$ of E as follows :

   for i=1,2,...,n :

   a. Find the set $J_i \subset \{1,2,\ldots,p\}$ defined as follows :

   $$J_i = \{j \in (1,2,\ldots,p) / d(x_i, x_j^k) = \min_{s=1,..,p} d(x_i, x_s^k)\}$$

   b. If $\text{card}(J_i)=1$, x_i is assigned to e_j^k, if not x_i is assigned to $e_{j_1}^k$ where $j_1 = \min_{j \in J_1} j$.

3. For i=1,2,...,p, compute x_j^{k+1}, inertia center of e_j^k (see(7)).
4. Test.

   4.1. If $x_j^{k+1} = x_j^k$ for j=1,2,...,p, end of the process.

   4.2. Else, set k=k+1 and go to 2.

The algorithm A1 converges.

Proof :

We define :

$I(y,E) = \sum_{x \in E} u_x d^2(y,x)$, where u_x is the weight of x and E a finite set.

If $x_j^{k+1} \neq x_j^k$ for at least one $j \in \{1,2,\ldots,p\}$, then :

$$\sum_{j=1}^{p} I(x_j^{k+1}, e_j^k) < \sum_{j=1}^{p} I(x_j^k, e_j^k) \qquad (8)$$

(see step 3 of the algorithm)

Furthermore :

$$\sum_{j=1}^{p} I(x_j^k, e_j^k) \leq \sum_{j=1}^{p} I(x_j^k, e_j^{k-1}) \qquad (9)$$

(see step 2 of the algorithm and the relation (7))

Finally, considering the relations (8) and (9) :

$$\sum_{j=1}^{p} I(x_j^{k+1}, e_j^k) < \sum_{j=1}^{p} I(x_j^k, e_j^{k-1}) \qquad (10)$$

This complete the proof. □

Using theorem II, we can easily obtain an algorithm which leads to a "good" partition of E starting from a set $x_1^0, x_2^0, \ldots, x_p^0$ of initial points.

We summarize the algorithm which leads to this set of points.

It consists in finding p points which are :

1. located in high density zones of $\mathbb{R}^m$ (i.e. in zones where are located a great number of points).
2. far one from the others.

V. AN EXAMPLE

The matrix to transform is given in figure 2. It concerns 60 part types and 12 tasks. The first column represents the part type numbers, and the last one contains the weights.

```
 1 0.1.0.1.0.0.0.0.0.0.0.1.3.
 2 1.0.0.0.1.0.0.0.0.1.1.0.3.
 3 0.0.0.1.0.1.0.0.1.0.0.0.1.
 4 1.0.0.0.0.1.0.0.1.1.0.0.2.
 5 1.1.0.1.0.0.0.1.0.0.0.1.2.
 6 0.0.0.0.0.0.0.1.0.0.0.0.2.
 7 0.1.0.0.0.0.1.0.1.1.0.1.2.
 8 0.0.0.0.0.0.0.0.0.1.0.1.4.
 9 1.0.0.0.0.0.0.0.0.0.1.1.1.
10 0.0.0.0.0.0.0.0.0.1.0.0.1.
11 0.0.0.0.0.0.0.0.1.0.0.0.3.
12 0.0.0.0.0.1.0.0.1.1.0.0.1.
13 0.1.0.1.0.0.0.1.0.0.0.0.1.
14 0.0.0.0.0.1.1.0.1.0.1.0.2.
15 0.1.0.0.0.0.0.1.0.0.0.0.1.
16 0.0.0.1.0.0.0.1.0.0.0.1.2.
17 0.1.0.0.0.0.1.0.0.1.0.1.1.
18 0.1.0.0.1.1.0.0.1.0.0.0.1.
19 0.1.0.0.0.1.0.1.0.0.0.0.5.
20 0.0.0.0.0.1.1.1.1.0.0.0.2.
21 0.0.0.0.1.0.1.1.0.1.0.1.4.
22 0.0.0.0.0.1.0.0.1.0.1.0.3.
23 1.0.1.0.0.0.1.0.1.0.0.1.3.
24 0.1.0.0.0.0.0.0.1.0.0.0.3.
25 0.0.1.0.1.0.1.0.0.0.0.0.2.
26 1.0.0.0.0.0.0.0.0.0.1.0.4.
27 0.1.1.0.1.0.1.0.0.0.0.0.2.
28 0.0.0.0.0.0.0.0.0.1.0.1.3.
29 0.0.1.0.0.0.1.0.1.1.1.0.3.
30 0.1.0.1.0.0.0.1.0.0.0.1.2.
31 0.0.0.1.0.0.0.1.0.0.0.0.1.
32 0.0.0.0.0.1.0.0.1.0.0.1.3.
33 0.1.0.1.0.0.0.1.0.0.0.0.1.
34 0.0.0.0.0.0.1.0.0.1.1.1.2.
35 0.0.1.0.1.0.0.0.0.0.0.0.1.
36 1.0.1.0.0.0.1.0.0.1.0.1.2.
37 0.0.0.0.0.0.0.1.0.1.0.0.4.
38 0.0.0.0.1.0.0.0.0.0.0.0.5.
39 0.0.0.0.0.1.0.0.0.0.0.0.1.
40 1.0.1.0.1.0.0.0.0.0.0.0.3.
41 1.0.0.0.0.1.0.0.0.0.1.0.3.
42 0.1.0.1.0.0.0.1.0.0.0.0.1.
43 0.0.1.0.1.0.0.0.0.1.0.0.2.
44 0.1.0.0.0.0.0.1.0.1.0.0.1.
45 0.0.0.1.0.1.0.1.0.0.0.0.1.
46 0.0.1.0.1.0.0.0.0.1.0.0.3.
47 0.1.0.0.0.0.0.1.0.0.0.0.1.
48 0.1.0.1.0.0.0.1.1.0.0.0.4.
49 0.0.0.0.0.0.1.0.0.1.0.0.1.
50 0.0.0.0.0.1.0.0.1.0.0.0.1.
51 0.1.0.1.0.0.0.0.0.0.0.1.3.
52 1.0.0.0.1.0.0.0.0.1.1.0.3.
53 0.0.0.1.0.1.0.0.1.0.0.0.1.
54 1.0.0.0.0.1.0.0.1.1.0.0.2.
55 1.1.0.1.0.0.0.1.0.0.0.1.2.
56 0.0.0.0.0.0.0.1.0.0.0.0.2.
57 0.1.0.0.0.0.1.0.1.1.0.1.2.
58 0.0.0.0.0.0.0.0.0.1.0.1.4.
59 1.0.0.0.0.0.0.0.0.0.1.1.1.
60 0.0.0.0.0.0.0.0.0.1.0.0.1.
```

Figure 2

The algorithm leads to the matrix of figure 3 if 5 part families are requested. The first column and the first row contain the class numbers. The second column contains the part type numbers.

The second row contains the column numbers of the initial matrix.

Value of the criterion : 1369

		1	1	1	2	2	2	3	3	4	4	5	5	
		8	4	2	12	10	7	9	6	5	3	11	1	
1	1	0.	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	3.
1	5	1.	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	1.	2.
1	6	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.
1	13	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	15	1.	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	16	1.	1.	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	2.
1	19	1.	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	0.	5.
1	30	1.	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	2.
1	31	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	33	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	42	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	44	1.	0.	1.	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.
1	45	1.	1.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	1.
1	47	1.	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	48	1.	1.	1.	0.	0.	0.	1.	0.	0.	0.	0.	0.	4.
1	51	0.	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	3.
1	55	1.	1.	1.	1.	0.	0.	0.	0.	0.	0.	0.	1.	2.
1	56	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.
2	34	0.	0.	0.	1.	1.	1.	0.	0.	0.	0.	1.	0.	2.
2	36	0.	0.	0.	1.	1.	1.	0.	0.	0.	1.	0.	1.	2.
2	37	1.	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	4.
2	7	0.	0.	1.	1.	1.	1.	1.	0.	0.	0.	0.	0.	2.
2	21	1.	0.	0.	1.	1.	1.	0.	0.	1.	0.	0.	0.	4.
2	23	0.	0.	0.	1.	0.	1.	1.	0.	0.	1.	0.	1.	3.
2	28	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	3.
2	29	0.	0.	0.	0.	1.	1.	1.	0.	0.	1.	1.	0.	3.
2	49	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	1.
2	8	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	4.
2	10	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.
2	17	0.	0.	1.	1.	1.	1.	0.	0.	0.	0.	0.	0.	1.
2	57	0.	0.	1.	1.	1.	1.	1.	0.	0.	0.	0.	0.	2.

2	58	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	4.
2	60	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.
3	11	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	3.
3	12	0.	0.	0.	0.	1.	0.	1.	1.	0.	0.	0.	0.	1.
3	18	0.	0.	1.	0.	0.	0.	1.	1.	1.	0.	0.	0.	1.
3	4	0.	0.	0.	0.	1.	0.	1.	1.	0.	0.	0.	1.	2.
3	32	0.	0.	0.	1.	0.	0.	1.	1.	0.	0.	0.	0.	3.
3	50	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	1.
3	20	1.	0.	0.	0.	0.	1.	1.	1.	0.	0.	0.	0.	2.
3	53	0.	1.	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	1.
3	54	0.	0.	0.	0.	1.	0.	1.	1.	0.	0.	0.	1.	2.
3	14	0.	0.	0.	0.	0.	1.	1.	1.	0.	0.	1.	0.	2.
3	22	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	1.	0.	3.
3	3	0.	1.	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	1.
3	39	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	1.
3	24	0.	0.	1.	0.	0.	0.	1.	0.	0.	0.	0.	0.	3.
4	38	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	5.
4	46	0.	0.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	3.
4	35	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	1.
4	40	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	1.	3.
4	27	0.	0.	1.	0.	0.	1.	0.	0.	1.	1.	0.	0.	2.
4	43	0.	0.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	2.
4	25	0.	0.	0.	0.	0.	1.	0.	0.	1.	1.	0.	0.	2.
5	26	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	4.
5	2	0.	0.	0.	0.	1.	0.	0.	0.	1.	0.	1.	1.	3.
5	52	0.	0.	0.	0.	1.	0.	0.	0.	1.	0.	1.	1.	3.
5	9	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	1.	1.	1.
5	59	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	1.	1.	1.
5	41	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	1.	1.	3.

Figure 3

The figure 4 gives the result obtained in the case of 10 part families are requested. Note that the algorithm leads to only 7 part families and production subsystems.

Value of the criterion : 1378

		2	2	3	3	4	4	5	6	9	9	10	10	
		12	10	4	2	5	3	7	8	11	1	9	6	
2	7	1.	1.	0.	1.	0.	0.	1.	0.	0.	0.	1.	0.	2.
2	8	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
2	10	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
2	17	1.	1.	0.	1.	0.	0.	1.	0.	0.	0.	0.	0.	1.
2	21	1.	1.	0.	0.	1.	0.	1.	1.	0.	0.	0.	0.	4.
2	28	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	3.
2	34	1.	1.	0.	0.	0.	0.	1.	0.	1.	0.	0.	0.	2.
2	36	1.	1.	0.	0.	0.	1.	1.	0.	0.	1.	0.	0.	2.
2	57	1.	1.	0.	1.	0.	0.	1.	0.	0.	0.	1.	0.	2.
2	58	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
2	60	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
3	5	1.	0.	1.	1.	0.	0.	0.	1.	0.	1.	0.	0.	2.
3	1	1.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	3.
3	42	0.	0.	1.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
3	48	0.	0.	1.	1.	0.	0.	0.	1.	0.	0.	1.	0.	4.
3	51	1.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	3.
3	55	1.	0.	1.	1.	0.	0.	0.	1.	0.	1.	0.	0.	2.
3	13	0.	0.	1.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
3	30	1.	0.	1.	1.	0.	0.	0.	1.	0.	0.	0.	0.	2.
3	33	0.	0.	1.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
4	46	0.	1.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	3.
4	35	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	1.
4	25	0.	0.	0.	0.	1.	1.	1.	0.	0.	0.	0.	0.	2.
4	38	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	5.
4	40	0.	0.	0.	0.	1.	1.	0.	0.	0.	1.	0.	0.	3.
4	27	0.	0.	0.	1.	1.	1.	1.	0.	0.	0.	0.	0.	2.
4	43	0.	1.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	2.
5	49	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	1.
5	29	0.	1.	0.	0.	0.	1.	1.	0.	1.	0.	1.	0.	3.
5	23	1.	0.	0.	0.	0.	1.	1.	0.	0.	1.	1.	0.	3.
6	6	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	2.

6	47	0.	0.	0.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
6	37	0.	1.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	4.
6	31	0.	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	0.	1.
6	19	0.	0.	0.	1.	0.	0.	0.	1.	0.	0.	0.	1.	5.
6	15	0.	0.	0.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
6	56	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	2.
6	16	1.	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	0.	2.
6	44	0.	1.	0.	1.	0.	0.	0.	1.	0.	0.	0.	0.	1.
6	45	0.	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	1.	1.
9	2	0.	1.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	3.
9	26	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	4.
9	41	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	1.	3.
9	9	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	1.
9	59	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	1.
9	52	0.	1.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	3.
10	12	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.
10	18	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	1.	1.	1.
10	3	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.
10	50	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.
10	39	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.
10	20	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	1.	1.	2.
10	53	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.
10	54	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.	2.
10	14	0.	0.	0.	0.	0.	0.	1.	0.	1.	0.	1.	1.	2.
10	22	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	1.	1.	3.
10	32	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	3.
10	4	0.	1.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.	2.
10	24	0.	0.	0.	1.	0.	0.	0.	0.	0.	0.	1.	0.	3.
10	11	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	3.

Figure 4

The last trial has been made with 15 requested part families. The figure 5 gives the results : finally, only 7 part families and production subsystems have been obtained.

Value of the criterion : 1371.

		1 12	1 10	2 8	2 4	4 11	4 1	5 7	8 9	8 6	11 2	15 5	15 3	
1	7	1.	1.	0.	0.	0.	0.	1.	1.	0.	1.	0.	0.	2.
1	8	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
1	10	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
1	17	1.	1.	0.	0.	0.	0.	1.	0.	0.	1.	0.	0.	1.
1	21	1.	1.	1.	0.	0.	0.	1.	0.	0.	0.	1.	0.	4.
1	28	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	3.
1	34	1.	1.	0.	0.	1.	0.	1.	0.	0.	0.	0.	0.	2.
1	36	1.	1.	0.	0.	0.	1.	1.	0.	0.	0.	0.	1.	2.
1	57	1.	1.	0.	0.	0.	0.	1.	1.	0.	1.	0.	0.	2.
1	58	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
1	60	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.
2	31	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	1.
2	33	0.	0.	1.	1.	0.	0.	0.	0.	0.	1.	0.	0.	1.
2	16	1.	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	2.
2	6	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.
2	37	0.	1.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
2	42	0.	0.	1.	1.	0.	0.	0.	0.	0.	1.	0.	0.	1.
2	45	0.	0.	1.	1.	0.	0.	0.	0.	1.	0.	0.	0.	1.
2	48	0.	0.	1.	1.	0.	0.	0.	1.	0.	1.	0.	0.	4.
2	55	1.	0.	1.	1.	0.	1.	0.	0.	0.	1.	0.	0.	2.
2	56	0.	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.
2	5	1.	0.	1.	1.	0.	1.	0.	0.	0.	1.	0.	0.	2.
2	13	0.	0.	1.	1.	0.	0.	0.	0.	0.	1.	0.	0.	1.
2	30	1.	0.	1.	1.	0.	0.	0.	0.	0.	1.	0.	0.	2.
4	41	0.	0.	0.	0.	1.	1.	0.	0.	1.	0.	0.	0.	3.
4	2	0.	1.	0.	0.	1.	1.	0.	0.	0.	0.	1.	0.	3.
4	26	0.	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	4.
4	9	1.	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	1.
4	59	1.	0.	0.	0.	1.	1.	0.	0.	0.	0.	0.	0.	1.

4	52	0.	1.	0.	0.	1.	1.	0.	0.	0.	0.	1.	0.	3.
5	29	0.	1.	0.	0.	1.	0.	1.	1.	0.	0.	0.	1.	3.
5	49	0.	1.	0.	0.	0.	0.	1.	0.	0.	0.	0.	0.	1.
5	23	1.	0.	0.	0.	0.	1.	1.	1.	0.	0.	0.	1.	3.
8	18	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.	1.	0.	1.
8	20	0.	0.	1.	0.	0.	0.	1.	1.	1.	0.	0.	0.	2.
8	32	1.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	0.	3.
8	50	0.	0.	0.	0.	0.	0.	0.	1.	1.	0.	0.	0.	1.
8	12	0.	1.	0.	0.	0.	0.	0.	1.	1.	0.	0.	0.	1.
8	53	0.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	0.	1.
8	54	0.	1.	0.	0.	0.	1.	0.	1.	1.	0.	0.	0.	2.
8	22	0.	0.	0.	0.	1.	0.	0.	1.	1.	0.	0.	0.	3.
8	3	0.	0.	0.	1.	0.	0.	0.	1.	1.	0.	0.	0.	1.
8	14	0.	0.	0.	0.	1.	0.	1.	1.	1.	0.	0.	0.	2.
8	39	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	1.
8	4	0.	1.	0.	0.	0.	1.	0.	1.	1.	0.	0.	0.	2.
8	11	0.	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	3.
11	47	0.	0.	1.	0.	0.	0.	0.	0.	0.	1.	0.	0.	1.
11	15	0.	0.	1.	0.	0.	0.	0.	0.	0.	1.	0.	0.	1.
11	24	0.	0.	0.	0.	0.	0.	0.	1.	0.	1.	0.	0.	3.
11	1	1.	0.	0.	1.	0.	0.	0.	0.	0.	1.	0.	0.	3.
11	51	1.	0.	0.	1.	0.	0.	0.	0.	0.	1.	0.	0.	3.
11	44	0.	1.	1.	0.	0.	0.	0.	0.	0.	1.	0.	0.	1.
11	19	0.	0.	1.	0.	0.	0.	0.	0.	1.	1.	0.	0.	5.
15	40	0.	0.	0.	0.	0.	1.	0.	0.	0.	0.	1.	1.	3.
15	27	0.	0.	0.	0.	0.	0.	1.	0.	0.	1.	1.	1.	2.
15	25	0.	0.	0.	0.	0.	0.	1.	0.	0.	0.	1.	1.	2.
15	43	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	2.
15	38	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	0.	5.
15	35	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	1.
15	46	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	1.	1.	3.

Main Program

```
 1 c
 2 c         programme mumu -> remplissage manuel du fichier 70
 3 c         programme alea -> remplissage aleatoire du fichier 70
 4 c         programme umum -> verification du contenue du fichier 70
 5 c
 6 c         le fichier 70 est deja rempli pour l'exemple demande (15,55)
 7 c         le fichier 50  -> resultats
 8 c
 9 c  CONTRAINTES ACTUELLES DU PROGRAMME
10 c----Nombre maxi. d'observations : 1999
11 c----Nombre maxi. de classes          : 50
12 c----Nombre maxi. de parametres       : 19
13        dimension a(50,20),aini(50,20),iz(2000),ds(50),it(500),b1(20),b2(20)
14       , dimension kz(20),jz(20),bj(50),kr(2000),kw(20)
15        open(70,access="direct",form="formatted",recl=2000)
16        open(63,access="direct",form="formatted",recl=2000)
17        open(50,form="formatted")
18        read(70,1,rec=1,err=27)n,m
19 c
20        eps=.001
21        write(0,70)n,m
22     70 format("_Nombre d'observations",i4,"  _Nombre de parametres",i4)
23 c
24      1 format(2i4)
25     21 write(0,5)
26      5 format(2x,"_Nombre de classes ? ")
27        read(0,3)ip
28      3 format(v)
29        write(0,200)
30    200 format(12x,"<< CHOIX DES DISTANCES POUR LA CONSTITUTION DES CLASSES
31       &>>",//)
32        write(0,201)
33    201 format(2x,"_Distance euclidienne ;.......taper 1 ",//)
34        read(0,3)idis
35        write(0,400)
36    400 format(12x,"<< CHOIX DES REPRESENTANTS >>")
37        write(0,401)
38    401 format(2x,"centre d'inertie; tapez 1")
39        read(0,3)irep
40 c
41 c
42 c                ****************************************
43 c                *       PHASE D'INITIALISATION         *
44 c                ****************************************
45 c
46 c                      Recherche du premier point.
47 c
48        write(0,2)
49      2 format(2x,"_Choix du rayon pour le calcul des densites. ")
50        read(0,3)r
51      8 write(0,4)
52      4 format(2x,"_Choix de la distance minimale entre points initiaux. ")
```

```
      read(0,3)dl
      if(dl.lt.r) goto 8
      call random_$uniform(x)
      if(x.gt.(1.-1.e-4))x=1.-1.e-4
      k=x*n+1
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(70,6,rec=k1,err=13)((a(k,j),j=1,m),k=1,k3)
    6 format(1000f2.0)
      jj=1
      do 7 j=1,m
      b1(j)=a(k2,j)
    7 aini(1,j)=a(k2,j)
c
c                  Recherche de la densite du premier
c                    point et des points suivants.
c
      do 12 j=1,50
   12 ds(j)=0
      ii=0
      do 9 k=1,n
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      if(k2.gt.1)go to 10
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(70,6,rec=k1,err=26)((a(i,j),j=1,m),i=1,k3)
   10 do 11 j=1,m
      b2(j)=a(k2,j)
   11 continue
      call dist(b1,b2,m,idis,y)
      if(y.le.r) ds(jj)=ds(jj)+1
      if(y.lt.dl) goto 9
      ii=ii+1
      it(ii)=k
    9 continue
c
c
c              ****************************************
c              *              ITERATIONS              *
c              ****************************************
c
c                        Choix du point suivant
c
   18 call rando._$uniform(x)
      if(x.gt.(1.-1.e-4))x=1.-1.e-4
      iil=x*ii+1
      k=it(iil)
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(70,6,rec=k1,err=28)((a(k,j),j=1,m),k=1,k3)
      jj=jj+1
      do 14 j=1,m
      b1(j)=a(k2,j)
      aini(jj,j)=a(k2,j)
   14 continue
```

```
c
c
c                 Recherche de la densite du point
c                    et des points a retenir.
c
      iil=0
      do 15 k5=1,ii
      k=it(k5)
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(70,6,rec=k1,err=29)((a(i,j),j=1,m),i=1,k3)
      do 17 j=1,m
   17 b2(j)=a(k2,j)
      call dist(b1,b2,m,idis,y)
      if(y.le.r) ds(jj)=ds(jj)+1
      if(y.lt.dl) goto 15
      iil=iil+1
      it(iil)=k
   15 continue
      ii=iil
      if(ii.gt.0) goto 18
      if(jj.ge.ip) goto 22
      write(0,19)
   19 format(2x,"_Nombre de points initiaux insuffisant .")
      write(0,20)
   20 format(2x,"_Abandon = 0",/,2x,"_Reprise = 1",/)
      read(0,3)is
      if(is.eq.0) goto 1664
      goto 21
c
c
c              ****************************************
c              *  CLASSEMENT DES POINTS INITIAUX      *
c              *            POSSIBLES                 *
c              ****************************************
c
c
   22 do 23 i=1,jj-1
      do 23 j=i+1,jj
      if(ds(i).gt.ds(j))goto 23
      do 24 k=1,m
      x=aini(i,k)
      aini(i,k)=aini(j,k)
   24 aini(j,k)=x
      x=ds(i)
      ds(i)=ds(j)
      ds(j)=x
   23 continue
c
c
c              ****************************************
c              *      FIN DE L'INITIALISATION         *
c              ****************************************
c
      index=0
c
c
c              ****************************************
```

```
c                *     CONSTITUTION DES CLASSES     *
c                ****************************************
c
c
c
  333 do 40 i=1,n
   40 iz(i)=0
      do 41 k=1,n
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      if(k2.gt.1) goto 42
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(70,6,rec=k1,err=39)((a(i5,j),j=1,m),i5=1,k3)
   42 do 43 j=1,m
   43 b1(j)=a(k2,j)
      do 44 i=1,ip
      do 45 j=1,m
   45 b2(j)=aini(i,j)
      call dist(b1,b2,m,idis,y)
      if(i.gt.1) goto 46
      z=y
      iz(k)=i
      goto44
   46 if(y.gt.z) goto 44
      z=y
      iz(k)=i
   44 continue
   41 continue
c
c
c                ****************************************
c                *    RECHERCHE DES REPRESENTANTS     *
c                ****************************************
c
c                Recherche des representants  suivants.
c
      call rep(ip,iz,aini,irep,n,m,idis,itel,eps)
      index=index+1
      if(index.lt.30) goto 600
      write(0,31)
   31 format(2x,"_Nombre d'iterations superieur a 30 .")
      goto 700
  600 if(itel.eq.1) goto 333
c
c                ************************************
c                * MAXIMISATION DU POIDS DU TABLEAU*
c                * **********************************
c
c                ************************************
c                * REGROUPEMENT DES OPERATIONS      *
c                ************************************
c
  700  pod=0.
       do 900 i=1,m-1
       do 900 j=1,ip
  900   aini(i,j)=0.
       do 901 k=1,n
       k1=(k-0.5)/50+2
       k2=k-(k1-2)*50
       if(k2.gt.1)go to 904
```

```
      k3=n-(k1-2)*50
      if(k3.gt.50)k3=50
      read(70,6,rec=k1,err=30)((a(i5,j),j=1,m),i5=1,k3)
904     do 905 j=1,ip
      if(iz(k).eq.j)go to 906
      do 907 i=1,m-1
907   aini(i,j)=aini(i,j)+(1-a(k2,i))*a(k2,m)
      go to 905
906   do 909 i=1,m-1
909   aini(i,j)=aini(i,j)+a(k2,i)*a(k2,m)
905   continue
901   continue
      do 910 i=1,m-1
      zz=0.
      nn=1
      do 911 j=1,ip
      if(aini(i,j).lt.zz)go to 911
      zz=aini(i,j)
      nn=j
911   continue
      pod=pod+zz
910   jz(i)=nn
      write(0,1115)pod
1115  format(5x,"Poids de la matrice :",f5.0)
c
c                ***********************************
c                *  REGROUPEMENT DES PRODUITS  *
c                ***********************************
c
      poc=0.
       do 921 k=1,n
      k1=(k-0.5)/50+2
      k2=k-(k1-2)*50
      if(k2.gt.1)go to 924
      k3=n-(k1-2)*50
      if(k3.gt.50)k3=50
      read(70,6,rec=k1,err=30)((a(i5,j),j=1,m),i5=1,k3)
924   do 928 j=1,ip
      bj(j)=0.
      do 75 ii=1,m-1
      if(jz(ii).eq.j) go to 76
75    continue
      go to 928
76    do 929 i=1,m-1
      if(jz(i).eq.j)go to 927
      bj(j)=bj(j)+(1-a(k2,i))*a(k2,m)
      go to 929
927   bj(j)=bj(j)+a(k2,i)*a(k2,m)
929   continue
928   continue
      zz=0.
      nn=1
      do 930 j=1,ip
      if(bj(j).lt.zz)go to 930
      zz=bj(j)
      nn=j
930   continue
      poc=poc+zz
921   iz(k)=nn
      write(0,1115)poc
```

```
      if(abs(poc-pod).gt.1)go to 700
c
c                        **********************
c                        *      EDITION       *
c                        **********************
c
c
c                    ************************************
c                    *  CREATION DU FICHIER FINAL       *
c                    ************************************
c
      write(50,1115)poc
      nn=(n-0.5)/50+2
      write(63,1,rec=1,err=27)n,m
      do 1012 k=1,n
 1012 kr(k)=k
      do 1013 i=1,n-1
      do 1013 j=i+1,n
      if((iz(j)-iz(i)).ge.0)go to 1013
      kk=iz(i)
      iz(i)=iz(j)
      iz(j)=kk
      kk=kr(i)
      kr(i)=kr(j)
      kr(j)=kk
 1013  continue
      do 1301 j=1,m
 1301 kw(j)=j
      do 1300 i=1,m-2
      do 1300 j=i+1,m-1
      if(jz(j).gt.jz(i))go to 1300
      kk=jz(i)
      jz(i)=jz(j)
      jz(j)=kk
      kk=kw(i)
      kw(i)=kw(j)
      kw(j)=kk
 1300 continue
      kk=0
      do 1020 ig=1,n
      k=kr(ig)
      k1=(k-0.5)/50+2
      k2=k-(k1-2)*50
      k3=n-(k1-2)*50
      if(k3.gt.50)k3=50
      read(70,6,rec=k1,err=30)((a(i5,j),j=1,m),i5=1,k3)
      kk=kk+1
      kk1=(kk-0.5)/50+2
      kk2=kk-(kk1-2)*50
      kk3=n-(kk1-2)*50
      if(kk3.gt.50)kk3=50
      do 1025 j=1,m
 1025  aini(kk2,j)=a(k2,j)
      if(kk2.ne.kk3)go to 1020
       write(63,6,rec=kk1,err=9999)((aini(i5,kw(j)),j=1,m),i5=1,k3)
 1020 continue
c
c
c                  ***************************************
c                  *       IMPRESSION DES RESULTATS      *
c                  ***************************************
```

```
      write(50,801)(jz(k),k=1,m-1)
      write(50,801)(kw(k),k=1,m-1)
 801  format(12x,20i3)
c
      do 702 k=1,n
      k1=(k-.5)/50+2
      k2=k-(k1-2)*50
      if(k2.gt.1) goto 704
      k3=n-(k1-2)*50
      if(k3.gt.50) k3=50
      read(63,6,rec=k1,err=8888)((a(i5,j),j=1,m),i5=1,k3)
 704  write(50,802)iz(k),kr(k),(a(k2,j),j=1,m)
802   format(2(2x,i4),2x,20(f2.0,1x))
 702  continue
 701  continue
      go to 1664
  30  write(0,32)
  32  format("erreur label 30")
      goto 1664
  29  write(0,33)
  33  format("erreur label 29")
      goto 1664
  28  write(0,34)
  34  format("erreur label 28")
      goto 1664
  26  write(0,35)
  35  format("erreur label 26")
      goto 1664
  27  write(0,36)
  36  format("erreur label 27")
      goto 1664
  13  write(0,37)
  37  format("erreur label 13")
      go to 1664
  39  write(0,38)
  38  format("erreur label 39")
      goto 1664
9999  write(0,3001)
3001  format(2x,"Erreur label 9999")
      goto 1664
8888  write(0,3002)
3002  format(2x,"Erreur label 8888")
1664  close(70)
      close(50)
      close(63)
      stop
      end
```

Subroutine rep

```
418 c
419 c
420 c
421       subroutine rep(ip,iz,aini,irep,n,m,idis,itel,eps)
422       dimension a(50,20),aini(50,20),iz(2000),b1(20),b2(20),co(50,20,3)
423       goto (1,100),irep
424     1 do 2 i=1,50
425       do 2,j=1,m-1
426       do 2 k=1,3
427     2 co(i,j,k)=0
428       do 12 i=1,50
429       do 12 j=1,20
430  12   aini(i,j)=0.
431       do 3 i=1,n
432       k1=(i-.5)/50+2
433       k2=i-(k1-2)*50
434       if(k2.gt.1) goto 4
435       k3=n-(k1-2)*50
436       if(k3.gt.50) k3=50
437 c
438       read(70,5,rec=k1,err=9)((a(k,j),j=1,m),k=1,k3)
439     5 format(1000f2.0)
440     4 i5=iz(i)
441       do 6 j=1,m-1
442       co(i5,j,1)=co(i5,j,1)+a(k2,m)
443       co(i5,j,2)=co(i5,j,2)-2*a(k2,m)*a(k2,j)
444     6 co(i5,j,3)=co(i5,j,3)+a(k2,m)*a(k2,j)**2
445     3 continue
446       itel=0
447       do 7 i=1,ip
448       if(co(i,1,1).gt.eps)go to 17
449       write(0,18)i
450  18   format(2x,"Representant num.",i4,"non utilise")
451       go to 7
452  17   do 7 j=1,m-1
453       a1=co(i,j,1)
454       a2=co(i,j,2)
455       a3=co(i,j,3)
456       pp=-a2/2./a1
457       uu=abs(pp-aini(i,j))
458       if(uu.gt.eps) itel=1
459       aini(i,j)=pp
460     7 continue
461       goto 1000
462   100 write(0,15)
463    15 format(2x,"_Coefficient de dissimilarite non prevu !")
464       go to 1000
465     9 write(*,8)
466     8 format(2x,"_Erreur lecture fichier70 dans sous-programme rep.")
467  1000 return
468       end
```

Subroutine dist

```
c
c
c
      subroutine dist(b1,b2,m,idis,y)
      dimension b1(20),b2(20)
      y=0
      goto (1,2),idis
    1 do 10 j=1,m-1
   10 y=y+(b1(j)-b2(j))**2
      y=y**.5
      goto 100
    2 do 20j=1,m-1
      i5=1
      i5=b1(j)+b2(j)
      if(i5.eq.2) i5=0
   20 y=y+i5
  100 return
      end
```

CONCLUSION

Number of trials have been made with the algorithm presented in this paper. Il seems to be effective and well adapted to the problems of great dimension.

The problem of the choise of the distance to use for finding the initial part families remains open. We used the euclidian distance for solving the above example. A systematic study of the available distances is expected.

A final remark : the approach proposed in this paper not only leads to the part families, but also gives the subsystems which are responsible of this classification. Using the Data Analysis language, this algorithm gives then the classes and, for each of them, the parameters which explain these classes.

BIBLIOGRAPHY

[1] ANDERBERG, M.R. (1973), "Cluster Analysis for Applications", Academic Press, New-York.

[2] GALLANGHER,C.C. and KNIGHT,W.A. (1973), "Group Technology", Butterworth and Co., London, U.K.

Imprimé en France

par

l'Institut National de Recherche en Informatique et en Automatique